马拉车算法

death00 发布于 2020-03-06

针对最长回文子串相关的问题，马拉车算法应该是比较通用的解法，今天我们就来具体看看这个算法。

简介

> 马拉车算法(Manacher's Algorithm)是用来查找一个字符串的最长回文子串的线性方法，由一个叫 Manacher 的人在1975年发明的，这个方法的最大贡献是在于将时间复杂度提升到了线性。

这个算法最厉害的地方是在于能够在线性时间内解决问题。一般我们解决最长回文子串，不可避免都要进行回溯之类的操作，那么时间复杂度一定是大于线性的。

而马拉车算法的主要思路是维护一个跟原字符串 str 长度一样的数组 lens，lens[i] 表示以 str[i] 为中点的回串其中一边的长度。

在这里，有的人把中点算进去，有的人记录两边的长度，其实都是一样的。我在这里是只记录一边的长度，不包括中点。比如CDCDE：

```
str:  [C, D, C, D, E]
lens: [0, 1, 1, 0, 0]
```

那么 lens 里最大的自然就对应最长回串的中点了。所以这个算法的核心就是如何快速计算 lens。

具体思路

预处理

回文有奇偶长度两种情况，通过补充间隔符可以将这两种情况化简为奇数长度。

比如：

> ABA补充为^#A#B#A#$，中点还是 B。
> ABBA补充为^#A#B#B#A#$，中点为 #，最后可以去掉。

针对间隔符，首先要确保在字符串中不会出现，这里我是确保字符串中不会出现^、#、$。

原字符串中每一个字符都会被#包围，这样就确保现在的字符串长度一定是奇数。

至于在开头增加^，在结尾增加$，这样是为了确保从任意一个位置开始检查回文时，一定会遇到不一样的时候，从而退出循环。而且也方便我们计算原字符的下标，直接除以2即可。

写法是：

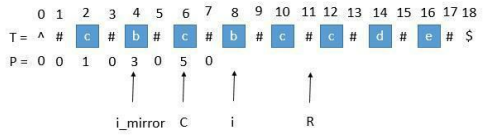
```
String str = "cbcbccde";
char[] T = new char[str.length() * 2 + 3];
T[0] = '^';
T[1] = '#';
T[T.length - 1] = '$';
for (int i = 0; i < str.length(); i++) {
    char charStr = str.charAt(i);
    T[2 * i + 2] = charStr;
    T[2 * i + 3] = '#';
}
```

计算长度数组

这就是算法的关键了，它充分利用了回文串的对称性。

我们用 C 表示回文串的中心，用 R 表示回文串的右边半径。所以 R = C + P[i]。C 和 R 所对应的回文串是当前循环中 R 最靠右的回文串。用 i_mirror 表示当前需要求的第 i 个字符关于 C 对应的下标。

让我们考虑求 P [i] 的时候：

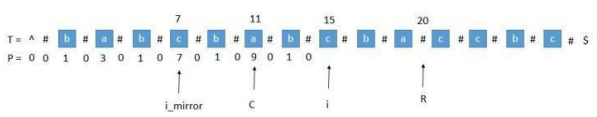


我们可以利用回文串 C 的对称性。i 关于 C 的对称点是 i_mirror，P [i_mirror] = 3，所以 P [i] 也等于 3 。

但需要考虑特殊情况：

P [i_mirror] + i >= R

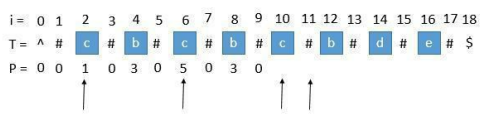
如下图：



当我们要求 P [i] 的时候，P [mirror] = 7，而此时 P [i] 并不等于 7，为什么呢，因为我们从 i 开始往后数 7 个，等于 22，已经超过了最右的 R，此时不能利用对称性了，但我们一定可以扩展到 R 的，所以 P [i] 至少等于 R - i = 20 - 15 = 5，会不会更大呢，我们只需要比较 T [R+1] 和 T [R+1] 关于 i 的对称点就行了，就像中心扩展法一样一个个扩展。

i_mirror - P [i_mirror] == 0

如下图：



death00
33 声望 3 粉丝
关注作者

文章目录 跟随

宣传栏

i_mirror C i R

此时P [i_mirror] = 1，但是 P [i] 赋值成 1 是不正确的，出现这种情况的原因是 P [i_mirror] 在扩展的时候首先是 "#" == "#"，之后遇到了 "^"和另一个字符比较，也就是到了边界，才终止循环的。而 P [i] 并没有遇到边界，所以我们可以继续通过中心扩展法一步一步向两边扩展就行了。

C 和 R 的更新

既然知道如何计算长度数组了，那最关键的 C 和 R 到底什么时候需要更新呢？

`i + P [ i ] > R`时，也就是当求出的 P [i] 的右边界大于当前的 R 时，我们就需要更新 C 和 R 为当前的回文串了。因为我们必须保证 i 在 R 里面，所以一旦有更右边的 R 就要更新 R。

最终写法

假设我们要写一个方法，传入参数是原字符串s，返回值是各个字符对应的最长回文子串长度数组，那么具体方法就是：

```
public int[] calSubstrings(String s) {
    if (s.length() == 0) {
        return new int[0];
    }
    // 存放新的内容
    char[] content = new char[s.length() * 2 + 3];
    // 开头用^
    content[0] = '^';
    // s中的每一个字符要用#包围
    content[1] = '#';
    for (int i = 0; i < s.length(); i++) {
        content[i * 2 + 2] = s.charAt(i);
        content[i * 2 + 3] = '#';
    }
    // 结尾用$
    content[content.length - 1] = '$';

    // 当前的回文串中心下标
    int center = 0;
    // 当前的回文串右边界
    int right = 0;
    // 存储以每一个位置为中心，所能获得的最长回文子串的长度
    int[] maxLength = new int[content.length];
    // 首尾两个字符没有必要计算
    for (int index = 1; index < content.length - 1; index++) {
        // 如果当前求解的位置，在右边界以内
```

总结

以上就是我关于马拉车算法的理解，用来解决最长回文子串的问题，简直就是一把利器。

有兴趣的话可以访问我的博客或者关注我的公众号、头条号，说不定会有意外的惊喜。

https://death00.github.io/

公众号：健程之道

算法

继续阅读

力扣227——227. 基本计算器 II

这道题类似于一般计算式解答，可以用栈解决，优化的时候可以利用题目本身的特殊性。原题 实现一个基本的计算器来计...

death00　阅读 868

【算法】排序算法

1.归并排序go语言实现demo_test.go {代码...}

一曲长歌一剑天涯　阅读 308

算法-图和图算法

可以用图对现实中的很多系统建模. 比如对交通流量建模, 顶点可以表示街道的十字路口, 边可以表示街道. 加权的边可以表...

伍陆柒　阅读 2.2k

FM算法和DeepFM算法

对于n个特征的模型，相比组合之前参数量级增加量级是n^2:n个特征组合后是n(n-1)/2，比如有(n=)1000个特征增加近5...

肖圣贤　阅读 5.5k

算法

递归冒泡

fubin　阅读 224

算法

目录 第1章　基础　1 1.1　基础编程模型　4 1.1.1　Java程序的基本结构　4 1.1.2　原始数据类型与表达式　6 1.1.3　语句　8...

ideaOzy　阅读 627

算法

1. 多数投票算法(Boyer-Moore Algorithm) 问题描述:给定一个无序数组，有n个元素，找出其中的一个多数元素，多数元素出...

l1nkkk　阅读 730